Logical proportions-related classification methods beyond analogy

Myriam Bounhas[1 2] Henri Prade[3]

[1] LARODEC Laboratory, ISG de Tunis, Tunisia
[2] Liwa College of Technology, Abu Dhabi, United Arab Emirates
[3] IRIT, CNRS & Univ. Paul Sabatier, Toulouse, France

SUM 2022, Paris, October 17-19, 2022

Outline

1. Introduction
2. Proposed approaches for classification
3. Link with (Ana)logical proportions
4. Experimental Validation
5. Conclusion

Table of contents

1. Introduction
2. Proposed approaches for classification
   - First Approach: Exploiting differences and Bongard problems
   - Second Approach: Using triplets of similar items
3. Link with (Ana)logical proportions
4. Experimental Validation
5. Conclusion

Introduction

Classification problem: different views

1. Sampling of an unknown probability distribution whose approximation governs the prediction of the class for a new item, *Cornuejols et al.(2020)* [5].

2. Logic-based approaches to classification, *Dubois and Prade (2020)* [6].
   - Example: *Boros et al. (2011)* [2] investigates the "justifiability" of rule-based classifiers.

Introduction

Classification problem: different views

1. Sampling of an unknown probability distribution whose approximation governs the prediction of the class for a new item, *Cornuejols et al.(2020)* [5].
2. Logic-based approaches to classification, *Dubois and Prade (2020)* [6].
   - Example: *Boros et al. (2011)* [2] investigates the "justifiability" of rule-based classifiers.

This paper: adopt the second alternative !

- Idea: Comparative reasoning between data.
- Option 1: Systematic analysis of the differences between the available examples.
- Option 2: Looks for similarities (beyond k-nearest neighbors methods) between input examples.

Table of contents

1 Introduction

2 Proposed approaches for classification
- First Approach: Exploiting differences and Bongard problems
- Second Approach: Using triplets of similar items

3 Link with (Ana)logical proportions

4 Experimental Validation

5 Conclusion

First Approach : Problem

- Input
  - a set of examples $\mathcal{E} = \{(\vec{x^i}, cl(\vec{x^i})) \mid i = 1, \cdots, m\}$,
  - $\vec{x^i} = (x_1^i, \cdots, x_j^i, \cdots, x_n^i)$ is a vector of n attributes of Boolean values.
  - $cl(\vec{x^i})$ denotes its class where $cl(\vec{x^i}) \in \{c_1, \cdots, c_{|C|}\})$.

First Approach : Problem

- Input
  - a set of examples $\mathcal{E} = \{(\vec{x^i}, cl(\vec{x^i})) \mid i = 1, \cdots, m\}$,
  - $\vec{x^i} = (x^i_1, \cdots, x^i_j, \cdots, x^i_n)$ is a vector of n attributes of Boolean values.
  - $cl(\vec{x^i})$ denotes its class where $cl(\vec{x^i}) \in \{c_1, \cdots, c_{|C|}\})$.
- Exploiting difference
  - Consider two examples $\vec{x^i}$ and $\vec{x^k}$.
  - Equal on a subset of attributes $Equ^{i,k} = \{j \mid x^i_j = x^k_j\}$
  - Differ on the subset $Dif^{i,k} = \{j \mid x^i_j \neq x^k_j\}$.

First Approach : Problem

- Input
  - a set of examples $\mathcal{E} = \{(\vec{x^i}, cl(\vec{x^i})) \mid i = 1, \cdots, m\}$,
  - $\vec{x^i} = (x^i_1, \cdots, x^i_j, \cdots, x^i_n)$ is a vector of n attributes of Boolean values.
  - $cl(\vec{x^i})$ denotes its class where $cl(\vec{x^i}) \in \{c_1, \cdots, c_{|C|}\})$.
- Exploiting difference
  - Consider two examples $\vec{x^i}$ and $\vec{x^k}$.
  - Equal on a subset of attributes $Equ^{i,k} = \{j \mid x^i_j = x^k_j\}$
  - Differ on the subset $Dif^{i,k} = \{j \mid x^i_j \neq x^k_j\}$.
- There are two cases:
  - If $cl(\vec{x^i}) = cl(\vec{x^k})$, it means that the difference between $\vec{x^i}$ and $\vec{x^k}$ observed on $Dif^{i,k}$ does not affect the class.
  - if $cl(\vec{x^i}) \neq cl(\vec{x^k})$, it means that the change in $Dif^{i,k}$ is enough for explaining the change from $cl(\vec{x^i})$ to $cl(\vec{x^k})$.

Exploiting differences and Bongard problems: Basic idea

Input:

- $\vec{d} \notin \mathcal{E}$: a new item s.t: $cl(\vec{d})$ is not known.
- Consider first the items that differ from $\vec{d}$ in only one attribute.
- Let $NN(\vec{d})$ be the set of these nearest neighbors of $\vec{d}$, and $\vec{c}$ be one of them.

Exploiting differences and Bongard problems: Basic idea

Input:

- $\vec{d} \notin \mathcal{E}$: a new item s.t: $cl(\vec{d})$ is not known.
- Consider first the items that differ from $\vec{d}$ in only one attribute.
- Let $NN(\vec{d})$ be the set of these nearest neighbors of $\vec{d}$, and $\vec{c}$ be one of them.

Strategy: Look at all the pairs $(\vec{a}, \vec{b}) \in \mathcal{E}^2$ s.t: $dif(\vec{a}, \vec{b}) = dif(\vec{c}, \vec{d})$ $(Equ^{\vec{a},\vec{b}} = Equ^{\vec{c},\vec{d}})$ to assess the effect of this difference:

Exploiting differences and Bongard problems: Basic idea

Input:

- $\vec{d} \notin \mathcal{E}$: a new item s.t: $cl(\vec{d})$ is not known.
- Consider first the items that differ from $\vec{d}$ in only one attribute.
- Let $NN(\vec{d})$ be the set of these nearest neighbors of $\vec{d}$, and $\vec{c}$ be one of them.

Strategy: Look at all the pairs $(\vec{a}, \vec{b}) \in \mathcal{E}^2$ s.t: $dif(\vec{a}, \vec{b}) = dif(\vec{c}, \vec{d})$ $(Equ^{\vec{a},\vec{b}} = Equ^{\vec{c},\vec{d}})$ to assess the effect of this difference:

- **Case 1:** $\forall(\vec{a}, \vec{b})$ s.t: $dif(\vec{a}, \vec{b}) = dif(\vec{c}, \vec{d})$, $cl(\vec{a}) = cl(\vec{b})$
  - Expect $cl(\vec{d}) = cl(\vec{c})$ according to the considered $\vec{c}$;
- **Case 2:** $\forall(\vec{a}, \vec{b})$ we have $cl(\vec{a}) \neq cl(\vec{b})$
  - Predict $cl(\vec{d}) = cl(\vec{b})$ according to the considered $\vec{c}$ if $cl(\vec{c}) = cl(\vec{a})$.
- **Case 3:** Conflict ! Two non-empty sets of pairs:
  - Set $S_1^{=}$: pairs s.t: $cl(\vec{a}) = cl(\vec{b})$.
  - Set $S_2^{\neq}$: pairs such that $cl(\vec{a}) \neq cl(\vec{b})$.

First Approach: Solving Bongard problem

Why the same change inside the pairs leads to different classes or not for the items in the pairs ?

First Approach: Solving Bongard problem

Why the same change inside the pairs leads to different classes or not for the items in the pairs ?

Proposed solution: Bongard problem, Bongard (1967) [1]

Look for a property P that is :

- True in the context of the pairs where $cl(\vec{a}) = cl(\vec{b})$
- False for the pairs such that $cl(\vec{a}) \neq cl(\vec{b})$.

First Approach: Solving Bongard problem

Why the same change inside the pairs leads to different classes or not for the items in the pairs ?

Proposed solution: Bongard problem, Bongard (1967) [1]

Look for a property P that is :

- True in the context of the pairs where $cl(\vec{a}) = cl(\vec{b})$
- False for the pairs such that $cl(\vec{a}) \neq cl(\vec{b})$.
- If a solution P exist:
  - If $\vec{d}$ has property/ies P then $cl(\vec{d}) = cl(\vec{c})$ for this $\vec{c}$;
  - otherwise $cl(\vec{d}) = cl(\vec{b})$ for this $\vec{c}$ if $cl(\vec{c}) = cl(\vec{a})$.
- If no solution P can be found, select another $\vec{c} \in NN(\vec{d})$.

First Approach: Solving Bongard problem

Why the same change inside the pairs leads to different classes or not for the items in the pairs ?

Proposed solution: Bongard problem, Bongard (1967) [1]

Look for a property P that is :

- True in the context of the pairs where $cl(\vec{a}) = cl(\vec{b})$
- False for the pairs such that $cl(\vec{a}) \neq cl(\vec{b})$.
- If a solution P exist:
  - If $\vec{d}$ has property/ies P then $cl(\vec{d}) = cl(\vec{c})$ for this $\vec{c}$;
  - otherwise $cl(\vec{d}) = cl(\vec{b})$ for this $\vec{c}$ if $cl(\vec{c}) = cl(\vec{a})$.
- If no solution P can be found, select another $\vec{c} \in NN(\vec{d})$.

Finally: apply a vote on the predictions made by successful neighbors.

Bongard problem: Example

Abstract dataset Or2 (7 Boolean Attributes): Or2 : $cl(x) = x_1 OR x_2$

A small Sample from "Or" Dataset		
	$S_1^{=}$	$S_2^{\neq}$
Pairs (a, b)	a: 1,1,1,0,0,1,1, cl=1 b: 0,1,1,0,0,1,1, cl=1 a: 1,1,1,1,1,1,0, cl=1 b: 0,1,1,1,1,1,0, cl=1 a: 1,1,0,0,1,0,0, cl=1 b: 0,1,0,0,1,0,0, cl=1 a: 1,1,1,1,0,0,0, cl=1 b: 0,1,1,1,0,0,0, cl=1	a: 1,0,0,0,0,0,0, cl=1 b: 0,0,0,0,0,0,0, cl=0 a: 1,0,0,1,1,1,0, cl=1 b: 0,0,0,1,1,1,0, cl=0 a: 1,0,1,0,1,1,0, cl=1 b: 0,0,1,0,1,1,0, cl=0 a: 1,0,0,0,0,0,1, cl=1 b: 0,0,0,0,0,0,1, cl=0 a: 1,0,1,0,1,1,1, cl=1 b: 0,0,1,0,1,1,1, cl=0
Property P	**P: (Attribute2 = 1)**	**$\overline{P}$: (Attribute 2 = 0)**

Figure: Illustrative example on How to solve a Bongard problem?

Bongard problem: Example

Abstract dataset Or2 (7 Boolean Attributes): Or2 : $cl(x) = x_1 OR x_2$

A small Sample from "Or" Dataset		
	$S_1^=$	$S_2^{\neq}$
Pairs (a, b)	a: 1,1,1,0,0,1,1, cl=1 b: 0,1,1,0,0,1,1, cl=1 a: 1,1,1,1,1,1,0, cl=1 b: 0,1,1,1,1,1,0, cl=1 a: 1,1,0,0,1,0,0, cl=1 b: 0,1,0,0,1,0,0, cl=1 a: 1,1,1,1,0,0,0, cl=1 b: 0,1,1,1,0,0,0, cl=1	a: 1,0,0,0,0,0,0, cl=1 b: 0,0,0,0,0,0,0, cl=0 a: 1,0,0,1,1,1,0, cl=1 b: 0,0,0,1,1,1,0, cl=0 a: 1,0,1,0,1,1,0, cl=1 b: 0,0,1,0,1,1,0, cl=0 a: 1,0,0,0,0,0,1, cl=1 b: 0,0,0,0,0,0,1, cl=0 a: 1,0,1,0,1,1,1, cl=1 b: 0,0,1,0,1,1,1, cl=0
Property P	**P: (Attribute2 = 1)**	**$\overline{\text{P}}$: (Attribute 2 = 0)**

Figure: Illustrative example on How to solve a Bongard problem?

Second Approach: Using triplets of similar items

Very simple strategy !

$\Rightarrow$ Consider triplets instead of pairs.

- **Input:** a set of examples $\mathcal{E} = \{(\vec{x^i}, cl(\vec{x^i})) \mid i = 1, \cdots, m\}$,
- **Goal:** Predict $\vec{d} \notin \mathcal{E}$: a new item s.t: $cl(\vec{d})$ is not known.

Second Approach: Using triplets of similar items

Very simple strategy !

$\Rightarrow$ Consider triplets instead of pairs.

- **Input:** a set of examples $\mathcal{E} = \{(\vec{x^i}, cl(\vec{x^i})) \mid i = 1, \cdots, m\}$,
- **Goal:** Predict $\vec{d} \notin \mathcal{E}$: a new item s.t: $cl(\vec{d})$ is not known.

Methodology:

1. Partition $\mathcal{E}$ into sets $\mathcal{C}$ of examples with the same label l, l is the class label of the set $\mathcal{C}$.
2. Compute $Equ(\vec{a}, \vec{b}, \vec{c}) = \{j \mid a_j = b_j = c_j\}$
3. Select only triplets t_s with high number of equal attributes i.e., $|Equ(\vec{a}, \vec{b}, \vec{c})| \geq \theta * NumberOfAttributes$ (θ is a fixed threshold).
4. For each of these triplets t_s, if $\vec{d}$ agree with t_s on the same attributes, increment the score for this class.
5. Allocate to $\vec{d}$ the class with the highest score.

Table of contents

1 Introduction

2 Proposed approaches for classification
- First Approach: Exploiting differences and Bongard problems
- Second Approach: Using triplets of similar items

3 Link with (Ana)logical proportions

4 Experimental Validation

5 Conclusion

Analogical proportions

Definition

- "a is to b as c is to d"
  a *differs* from b as c *differs* from d
  and b *differs* from a as d *differs* from c"

Analogical proportions

Definition

- "*a* is to *b* as *c* is to *d*"
  a differs from *b* as *c differs* from *d*
  and *b differs* from *a* as *d differs* from *c*"
- $a : b :: c : d \triangleq$
  $((a \wedge \neg b) \equiv (c \wedge \neg d)) \wedge ((\neg a \wedge b) \equiv (\neg c \wedge d))$

Analogical proportions

Definition

- "*a is to b as c is to d*"
  a differs from *b* as *c differs* from *d*
  and *b differs* from *a* as *d differs* from *c*"
- $a : b :: c : d \triangleq$
  $((a \wedge \neg b) \equiv (c \wedge \neg d)) \wedge ((\neg a \wedge b) \equiv (\neg c \wedge d))$

- $a : b :: c : d$ is true only for 6 valuations:
  $(a, b, c, d) \in \{(0, 0, 0, 0), (1, 1, 1, 1), (0, 0, 1, 1), (1, 1, 0, 0), (1, 0, 1, 0), (0, 1, 0, 1)\}$

Analogical proportions

Definition

- "a is to b as c is to d"
  a *differs* from b as c *differs* from d
  and b *differs* from a as d *differs* from c"
- $a : b :: c : d \triangleq$
  $((a \wedge \neg b) \equiv (c \wedge \neg d)) \wedge ((\neg a \wedge b) \equiv (\neg c \wedge d))$

- $a : b :: c : d$ is true only for 6 valuations:
  $(a, b, c, d) \in \{(0,0,0,0), (1,1,1,1), (0,0,1,1), (1,1,0,0), (1,0,1,0), (0,1,0,1)\}$
- AP between vectors: $\vec{a} : \vec{b} :: \vec{c} : \vec{d}$ if $\forall i = 1, \ldots, n,\ a_i : b_i :: c_i : d_i$

Analogical proportions

Definition

- "a is to b as c is to d"
  a *differs* from b as c *differs* from d
  and b *differs* from a as d *differs* from c"
- $a : b :: c : d \triangleq$
  $((a \wedge \neg b) \equiv (c \wedge \neg d)) \wedge ((\neg a \wedge b) \equiv (\neg c \wedge d))$

- $a : b :: c : d$ is true only for 6 valuations:
  $(a, b, c, d) \in \{(0,0,0,0), (1,1,1,1), (0,0,1,1), (1,1,0,0), (1,0,1,0), (0,1,0,1)\}$
- AP between vectors: $\vec{a} : \vec{b} :: \vec{c} : \vec{d}$ if $\forall i = 1, \ldots, n,\ a_i : b_i :: c_i : d_i$
- Analogical Inference $a : b :: c : x$ may not have a solution in $\mathbb{B}$
  neither $0 : 1 :: 1 : x$ nor $1 : 0 :: 0 : x$ have a solution
  - when it exists (iff $(a \equiv b) \vee (a \equiv c)$ holds) it is unique

Inverse paralogy

Bongard problems are related to "Inverse Paralogy" (IP).

Definition

- A quaternary logical connective: "what *a* and *b* have in common *c* and *d* have not it in common, and vice versa".

Inverse paralogy

Bongard problems are related to "Inverse Paralogy" (IP).

Definition

- A quaternary logical connective: "what a and b have in common c and d have not it in common, and vice versa".
- IP(a, b, c, d)= $[(a \wedge b) \equiv (\neg c \wedge \neg d)] \wedge [(\neg a \wedge \neg b) \equiv (c \wedge d)]$.

- $IP(a, b, c, d)$ is true only for 6 valuations:

  $(a, b, c, d) \in \{(1, 1, 0, 0), (0, 0, 1, 1), (0, 1, 1, 0), (1, 0, 0, 1), (1, 0, 1, 0), (0, 1, 0, 1)\}$

Baseline Analogical Classifier

A brute force **AP-classifier** *Bounhas et al. (2017)* [3] for comparison:

1. Look for each triplet $(\vec{a}, \vec{b}, \vec{c})$ in the example set.
2. Solve $cl(\vec{a}) : cl(\vec{b}) :: cl(\vec{c}) : y$.
3. If the previous analogical equation on classes has a solution l and if the analogical equation on the attributes is valid, i.e., $\vec{a} : \vec{b} :: \vec{c} : \vec{d}$, increase $score(l)$ by 1.
4. Assign to $\vec{d}$ the class label having the highest score as $cl(\vec{d}) = argmax_l(score(l))$

Table of contents

1 Introduction

2 Proposed approaches for classification
- First Approach: Exploiting differences and Bongard problems
- Second Approach: Using triplets of similar items

3 Link with (Ana)logical proportions

4 Experimental Validation

5 Conclusion

Experimental Validation

Datasets:

1. **8 Abstract Boolean functions (7 attributes):**
   - And2 : $cl(x) = x_1 AND x_2$
   - Or2 : $cl(x) = x_1 OR x_2$
   - Not : $cl(x) = x_1 ANDNot(x_2)$
   - And7 : $cl(x) = x_1 AND ... AND x_7$
   - Or7 : $cl(x) = x_1 OR ... OR x_7$
   - XOR : $cl(x) = x_1 XOR x_2$
   - XORMin : $cl(x) = x_1 XOR x_2, if(Sum(x_1, ..., x_7) < 6)$
     $Min(x_1, ..., x_7), otherwise$
   - Sum7 : $cl(x) = Sum(x1, ..., x7) = 2$

Experimental Validation

Datasets:

1. **8 Abstract Boolean functions (7 attributes):**
   - And2 : $cl(x) = x_1 AND x_2$
   - Or2 : $cl(x) = x_1 OR x_2$
   - Not : $cl(x) = x_1 AND Not(x_2)$
   - And7 : $cl(x) = x_1 AND ... AND x_7$
   - Or7 : $cl(x) = x_1 OR ... OR x_7$
   - XOR : $cl(x) = x_1 XOR x_2$
   - XORMin : $cl(x) = x_1 XOR x_2, if (Sum(x_1, ..., x_7) < 6)$ $Min(x_1, ..., x_7), otherwise$
   - Sum7 : $cl(x) = Sum(x1, ..., x7) = 2$
2. **U.C.I. Machine Learning repository [7]**
   - Binary classes databases: `Monk1`, `Monk2`, `Monk3` and `TicTacToe`.
   - Multiple classes databases: `Balance`, `Car` and `Hayes-Roth`.

Experimental Validation

Table: Description of datasets

Datasets	Ins.	Nom. Att.	Bin. Att.	Nb class	Def.: $cl(\vec{x}) =$
And2	128	-	7	2	$x_1 AND x_2$
Or2	128	-	7	2	$x_1 OR x_2$
Not	128	-	7	2	$x_1 AND Not(x_2)$
And7	128	-	7	2	$x_1 AND ... AND x_7$
Or7	128	-	7	2	$x_1 OR ... OR x_7$
XOR	128	-	7	2	$x_1 XOR x_2$
XORMin	128	-	7	2	$x_1 XOR x_2$, *if* $(Sum(x_1, ..., x_7) < 6)$ $Min(x_1, ..., x_7)$, *otherwise*
Sum7	128	-	7	2	$Sum(x1, ..., x7) = 2$
Monk1	432	6	15	2	-
Monk2	432	6	15	2	-
Monk3	432	6	15	2	-
TicTacToe	521	9	27	2	-
Balance	625	4	20	3	-
Car	743	7	21	4	-
Hayes-Roth	132	5	15	3	-

Experimental Validation

Testing strategy

- Boolean functions: Random samplings of 7 Boolean variables.
- U.C.I. ML datasets: All nominal attributes have been binarized using the free Weka software.
- Standard 10-fold cross-validation.
- Average accuracies over the 10 different values.

Experimental Validation

Testing strategy

- Boolean functions: Random samplings of 7 Boolean variables.
- U.C.I. ML datasets: All nominal attributes have been binarized using the free Weka software.
- Standard 10-fold cross-validation.
- Average accuracies over the 10 different values.
- Again: an Inner cross-validation for parameter optimization using the $\mathcal{E}$ only.
  - Algo1 : $k = 1{,}3{,}5, 7$
  - Algo2 : $\theta = 0.5{,}0.6{,}0.7{,}0.8$
  - kNN : $k = 1{,}2, ..., 11$
- Best parameter for each classifier is used for predicting the label for testing examples.

Results

Table: Accuracy results (means and standard deviations)

Dataset	Algo1		Algo2		BaselineAC	kNN		*Odd*3
		k^*		θ^*			k^*	[4]
And2	**100**	1	**100**	0.5	**100**	99.53 ± 1,08	1	-
Or2	**100**	1	99.84 ± 0.46	0.5	**100**	**100**	1	-
Not	**100**	1	**100**	0.5	**100**	99.69 ± 0.92	1	-
And7	98.28 ± 3,96	3	96.88 ± 4.60	0.6	96.88 ± 4.60	**99.69** ± **0.92**	1	-
Or7	**98.44** ± **3,41**	3	**98.44** ± **3.41**	0.6	**98.44** ± **3.41**	**98.44** ± **3.10**	1	-
XOR	**100**	1	**100**	0.5	**100**	99.38 ± 1.58	3	-
XORMin	96.41 ± 4.70	3	96.72 ± 4.39	0.5	**96.88** ± **4,08**	93.75 ± 6.29	1	-
Sum7	**99.06** ± **2.49**	5	83.59 ± 8.28	0.5	82.03 ± 8.27	82.50 ± 10.92	8	-
Monk1	**100**	1	**100**	0.7	99.95 ± 0.14	99.95 ± 0.14	3	99.31±3.39
Monk2	**100**	1	67.13 ± 6.14	0.5	99.54 ± 0.82	64.44 ± 6.99	11	60.93±4.16
Monk3	**100**	1	**100**	0.7	97.36 ± 1.78	**100**	1	99.95±0.05
TicTacToe	**100**	1	97.50 ± 2.28	0.7	**100**	98.27 ± 1.77	1	-
Balance	**95.36** ± **2.59**	7	89.84 ± 3.06	0.5	90.05 ± 3.35	83.94 ± 4.23	11	88.62±3.4
Car	**95.33** ± **2.40**	3	94.03 ± 3.03	0.8	91.22 ± 3.23	92.33 ±3.10	1	90.93±4.03
Hayes-Roth	80.30 ± 10.65	3	76.71 ± 12.74	0.5	**80.45** ± **9.22**	61.36 ± 13.46	3	79.37±9.74
Average	**97.54**	-	**93.37**	-	**95.5**	**91.5**	-	-

Main results

1. **Synthetic Boolean functions :**
   - Algo 1: best classifier (average accuracy) for all datasets except *And7* function (kNN : best for this dataset).
   - Algo 1: largely better than Algo2, the BaselineAC and kNN (optimized k) for dataset *Sum7* (all classifiers achieve an accuracy of about 80% while Algo1 achieves 99%).
   - Algo2: as good as or performs significantly better than the BaselineAC for most datasets (see the Sum7 dataset).

Main results

1. **Synthetic Boolean functions :**
   - Algo 1: best classifier (average accuracy) for all datasets except *And7* function (kNN : best for this dataset).
   - Algo 1: largely better than Algo2, the BaselineAC and kNN (optimized k) for dataset *Sum7* (all classifiers achieve an accuracy of about 80% while Algo1 achieves 99%).
   - Algo2: as good as or performs significantly better than the BaselineAC for most datasets (see the Sum7 dataset).
2. **U.C.I. ML datasets :**
   - Algo 1: very good accuracy for binary or multiple class problems.
   - Algo 1: significantly outperforms the kNN and the BaselineAC (Monk3, Balance and Car).
   - Overall Algo 2 is significantly better than the k-NN for most datasets.
   - As expected: better results for small values of k ($k = 1$) and θ ($\theta = 0.5$) for most datasets.

Results : Main conclusions

Why Algo 1 is better than to Algo 2 (and also k-NN) ?

Results : Main conclusions

Why Algo 1 is better than to Algo 2 (and also k-NN) ?

- Algo 1 uses examples with patterns of types $s : s :: t : y$ and $s : t :: s : y$ (same/different classes).
- Algo 2 exploits only patterns of type $s : s :: s : y$ (Restriction !).

Results : Main conclusions

Why Algo 1 is better than to Algo 2 (and also k-NN) ?

- Algo 1 uses examples with patterns of types $s : s :: t : y$ and $s : t :: s : y$ (same/different classes).
- Algo 2 exploits only patterns of type $s : s :: s : y$ (Restriction !).
- Algo 1 : Neighborhood of the item to be classified, quite different from the classical *kNN*.
  - Classify $\vec{d}$ as its kNN $\vec{c}$ only if for *all* pairs $(\vec{a}, \vec{b}) \in \mathcal{E}^2$ s.t: $dif(\vec{a}, \vec{b}) = dif(\vec{c}, \vec{d})$, $cl(\vec{a}) = cl(\vec{b})$.
  - Classify $\vec{d}$ as item(s) $\vec{b}$ (not as $\vec{c}$) if for *all* pairs $(\vec{a}, \vec{b}) \in \mathcal{E}^2$, $cl(\vec{a}) \neq cl(\vec{b})$.

Results : Main conclusions

Why Algo 1 is better than to Algo 2 (and also k-NN) ?

- Algo 1 uses examples with patterns of types $s : s :: t : y$ and $s : t :: s : y$ (same/different classes).
- Algo 2 exploits only patterns of type $s : s :: s : y$ (Restriction !).
- Algo 1 : Neighborhood of the item to be classified, quite different from the classical *kNN*.
  - Classify $\vec{d}$ as its kNN $\vec{c}$ only if for *all* pairs $(\vec{a}, \vec{b}) \in \mathcal{E}^2$ s.t: $dif(\vec{a}, \vec{b}) = dif(\vec{c}, \vec{d})$, $cl(\vec{a}) = cl(\vec{b})$.
  - Classify $\vec{d}$ as item(s) $\vec{b}$ (not as $\vec{c}$) if for *all* pairs $(\vec{a}, \vec{b}) \in \mathcal{E}^2$, $cl(\vec{a}) \neq cl(\vec{b})$.

Conclusion

Algo 1 applies a deeper investigation on the relationship between the change in attribute values that may affect/not affect the change in the class label.

Results : Bongard Problem

Strategy:

Compute the average proportion of examples that has been classified by solving Bongard problems

$$P_{Bongard} = \frac{\textit{Nbr of classified examples by solving Bongard pbms*100}}{\textit{Nbr of classified examples}}$$

Results : Bongard Problem

Strategy:

Compute the average proportion of examples that has been classified by solving Bongard problems

$$P_{Bongard} = \frac{\textit{Nbr of classified examples by solving Bongard pbms*100}}{\textit{Nbr of classified examples}}$$

Results:

- For some datasets (XOR), no Bongard problem has been solved: $P_{Bongard} = 0$ (case 1 or case 2 and never case 3 !) $\Rightarrow$ No prediction error!
- And2, Or2, Not, Car: $P_{Bongard} > 20\%$ (case 3) $\Rightarrow$ Still no prediction error !
- Monk1, Monk2, Monk3, Hayes-Roth: $P_{Bongard} > 1\%$

Note:

- If solving the Bongard problem lead to no property, $\vec{d}$ remains unclassified according to this NN $\vec{c}$.
- Algo1 passes to another nearest neighbor.

Table of contents

1 Introduction

2 Proposed approaches for classification
- First Approach: Exploiting differences and Bongard problems
- Second Approach: Using triplets of similar items

3 Link with (Ana)logical proportions

4 Experimental Validation

5 Conclusion

Conclusion

- The paper investigates two new classification methods using comparative reasoning.
- Processing of pairs or triplets of examples rather than individual examples (CBR).

Conclusion

- The paper investigates two new classification methods using comparative reasoning.
- Processing of pairs or triplets of examples rather than individual examples (CBR).
- First Approach: exploits pairs and appears especially efficient.

Comparative reasoning of pairs of examples: Strategy:

- Pairs $S_1^{=}$: The change in attribute values causes no effect on the class.
- Pairs $S_2^{\neq}$: The change in attribute values causes a change of class.
- Ambiguity of comparative reasoning result $\Rightarrow$ Solve Bongard problems for separating the contexts.

Conclusion

- The paper investigates two new classification methods using comparative reasoning.
- Processing of pairs or triplets of examples rather than individual examples (CBR).
- First Approach: exploits pairs and appears especially efficient.

Comparative reasoning of pairs of examples: Strategy:

- Pairs $S_1^{=}$: The change in attribute values causes no effect on the class.
- Pairs $S_2^{\neq}$: The change in attribute values causes a change of class.
- Ambiguity of comparative reasoning result $\Rightarrow$ Solve Bongard problems for separating the contexts.

The first approach outperforms the BaselineAC and k-NN.

Future directions

Extensions ?

1. Extend the experimentation to a variety of datasets.

Future directions

Extensions ?

1. Extend the experimentation to a variety of datasets.
2. Extension of the first procedure to deal with nominal attributes.

Future directions

Extensions ?

1. Extend the experimentation to a variety of datasets.
2. Extension of the first procedure to deal with nominal attributes.
3. First approach: Simple and natural procedure that may help for explanation:

   What are the properties/attributes that are responsible for the change in class label?

Thank you !

References I

M. M. Bongard.
Pattern Recognition.
Hayden Book, Spartan Books, Rochelle Park, 1970.
Russian version, Nauka Press, Moscow, 1967.

E. Boros, Y. Crama, P. L. Hammer, T. Ibaraki, A. Kogan, and K. Makino.
Logical analysis of data: classification with justification.
Annals of Oper. Res., 188(1):33–61, 2011.

M. Bounhas, H. Prade, and G. Richard.
Analogy-based classifiers for nominal or numerical data.
Int. J. of Approximate Reasoning, 91:36 – 55, 2017.

M. Bounhas, H. Prade, and G. Richard.
Oddness-based classification: A new way of exploiting neighbors.
Int. J. Intell. Syst., 33(12):2379–2401, 2018.

A. Cornuejols, F. Koriche, and R. Nock.
Statistical computational learning.
In P. Marquis, O. Papini, and H. Prade, editors, *A Guided Tour of Artificial Intelligence Research. 1 - Knowledge Representation, Reasoning and Learning*, pages 341–388. Springer-Verlag, 2020.

D. Dubois and H. Prade.
Towards a logic-based view of some approaches to classification tasks.
In M.-J. Lesot, S. M. Vieira, M. Z. Reformat, J. P. Carvalho, A. Wilbik, B. Bouchon-Meunier, and R. R. Yager, editors, *Proc. 18th Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU'20), Part III, Lisbon, June 15-19*, volume 1239 of *Communications in Computer and Information Science*, pages 697–711. Springer, 2020.

References II

J. Mertz and P. M. Murphy.
Uci repository of machine learning databases.
Available at: ftp://ftp.ics.uci.edu/pub/machine-learning-databases, 2000.